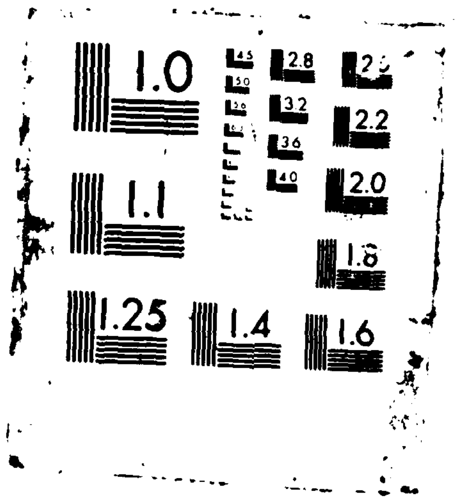
AD-A178 971 TKE EFFECT OF SOFTWARE SUPPORT NEEDS ON THE DEPARTMENT 1/1
OF DEFENSE SOFTWAR..(U) CARNEGIE-MELLON UNIV PITTSBURGH
PA SOFTWARE ENGINEERING INST.. A C MARTIN ET AL.
UNCLASSIFIED JAN 87 CMU/SEI-87-TR-2-PT-1 F/G 9/2 NL


1.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

Technical Report
CMU/SEI-87-TR-2
ESD-TR-87-102

2

Carnegie-Mellon University
Software Engineering Institute

The Effect of Software Support Needs on the Department of Defense Software Acquistion Policy: Part 1

i

A Framework for Analyzing Legal Issues

Anne C. Martin
Kevin M. Deasy

January 1987

Technical Report
ESD-TR-87-102
SEI-87-TR-2
January 1987

The Effect of Software Support Needs on the Department of Defense Software Acquisition Policy: Part 1 A Framework for Analyzing Legal Issues

Anne C. Martin
Kevin M. Deasy

Software Engineering Institute
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

This technical report was prepared for the

SEI Joint Program Office
ESD/XRS
Hanscom AFB, MA 01731

The ideas and findings in this report should not be construed as an official DoD position. It is published in the interest of scientific and technical information exchange.

Review and Approval

This report has been reviewed and is approved for publication.

FOR THE COMMANDER

Dan Burton

Daniel Burton
SEI Joint Program Office

This work was sponsored by the Department of Defense.

Table of Contents

A-1

The Effect of Software Support Needs on the Department of Defense Software Acquisition Policy: Part 1

A Framework for Analyzing Legal Issues

Anne C. Martin
Kevin M. Deasy

Abstract It is often in the context of acquiring intellectual property needed to maintain and enhance software that data rights disputes arise between DoD and private industry. DoD representatives identify access to source code, documentation and software tools as being vital to maintenance and enhancement. This report discusses technical and managerial variables that might affect DoD's need for such intellectual property. It is concluded that the choice of a group to carry out maintenance and enhancement is a key factor in determining DoD's requirements. Other important variables include use of software standards, higher order programming languages and reused software. Consideration of these factors could enable DoD to tailor more narrowly an acquisition to those items needed to maintain and enhance software, potentially reducing both DoD's costs, and the tension that exists between DoD and private industry with respect to data rights issues. [illegible]

PREFACE

Problem

The Department of Defense (DoD) has directed the Software Engineering Institute (SEI), through the latter's Software Licensing Project (the Project), to study issues relating to the acquisition of rights in software technology. As its initial task, the Project last year conducted a general investigation of DoD's software acquisition policy. That investigation culminated in a published report entitled "Toward a Reform of the Defense Department Software Acquisition Policy" [TR-1 86]. As a result of that effort, it became apparent that software and data rights acquisition problems relating to the maintenance and enhancement of software are critical areas of concern to the government contracts community. Accordingly, this year the Project is exploring legal issues arising from DoD's need to maintain software.

Approach

This investigation adopts a four-step approach.[1] This report summarizes the first step — investigation of technical and managerial issues related to maintaining software. The project's subsequent work will focus on potential legal issues stemming from software maintenance and ultimately recommend solutions to problems that may arise in technical and managerial areas. In conducting this investigation, the authors visited several DoD facilities involved in software acquisition and/or maintenance. Appendix A to this report lists those facilities. The authors also visited the Defense Intelligence Agency, the Institute for Defense Analyses, the Institute for Computer Sciences and Technology of the National Bureau of Standards, and the Federal Software Maintenance Group (FEDMAIN). In addition, defense contractors and members of the SEI technical staff were interviewed. All interviews were conducted in a formal question and answer format with anonymity assured to the interviewee due to the potentially controversial nature of some of the issues being discussed. The Project supplemented the field investigation with a review of the literature on software engineering and maintenance, focusing on some of the most respected scholars in these areas. A bibliography of relevant books, articles, and other documents is attached to this report as Appendix B. Members of the SEI technical staff have reviewed this report for accuracy.

[1]The second step of the project will focus on legal and policy constraints upon the software and data rights acquisition process. The third step will examine strategies DoD can use to obtain technology needed for software support within the framework of the technical, managerial and legal concerns. During this third step, recommendations will be made, and comment sought, on ways in which DoD software procurement policy can be improved. Finally, the fourth step will involve a refinement of these recommendations in light of comments obtained from industry and government representatives.

Scope of Report

This report summarizes the significant technical and managerial considerations that affect the maintenance and enhancement of software. Our prior work suggested that it is often in the acquisition of intellectual property needed to maintain and enhance software that data rights disputes arise between DoD and the private sector. For this reason, an understanding of DoD's maintenance and enhancement requirements is a necessary predicate toward shaping a data rights/software acquisition policy that achieves the proper balance between the intellectual property needs of DoD and the proprietary interests of private industry. A survey of software engineering literature revealed no study that addressed DoD maintenance and enhancement requirements as they relate to intellectual property needs. Accordingly, the Project undertook to examine the issue itself. Although this report discusses technical and managerial issues, it is principally intended as a guide for lawyers and policymakers who deal with, and have regulatory responsibility for, software and data rights acquisition issues.

INTRODUCTION

There is growing tension between DoD and private industry about the legitimate reach of DoD's rights in software technology under its current data rights policy [TaskForce 84, IDA 84]. DoD has generally claimed that it requires broad rights in software technology to meet its maintenance and enhancement needs. Developers have tended to view these DoD claims as overly broad and unjustified. Many private software developers note that software tools are often among the most innovative of technological developments, and may involve substantial private investment. Consequently, some contractors believe that DoD data rights policy causes developers to withhold some of their best technology from DoD, and discourages the development of innovative software development tools.

As noted by the President's Blue Ribbon Commission on Defense Management (the Packard Commission) [Packard 86, p. 64],

> "DoD must recognize the delicate and necessary balance between the government's requirement for technical data and the benefit to the nation that comes from protecting the private sector's proprietary rights. That balance must be struck so as to foster technological innovation and private investment which is so important in developing products vital to our defense."

In order to determine how such a "delicate and necessary balance" may be struck in the context of a software/data rights acquisition policy, this Project began by studying the characteristics of software maintenance and the technology it requires. As a result of this research, the Project reached four conclusions:

1. Due to the dynamic nature of software, the maintenance of software differs fundamentally from the maintenance of hardware. Maintaining software requires not only correcting defects, but it also requires enhancing the software to meet changing needs.
2. The maintenance and enhancement of software, as opposed to the maintenance of hardware, increases the need to transfer intellectual property pertaining to the software system from the developer to maintenance personnel.
3. The need to transfer intellectual property relating to software maintenance, and the technology in which it is embodied, may differ from system to system, depending on numerous variables. The variables include the method of software support and certain technological factors, e.g., the use of standardization, the use of higher order programming languages, and reused software.
4. Any informed decision regarding maintenance and enhancement requirements requires consultation with the personnel that support the system.

An appreciation of the type of technology needed to maintain software will aid DoD in assessing its data rights policy. This appreciation will help DoD to identify those situations in which it should acquire rights in technology as well as those situations in which rights in technology are neither necessary nor desirable.

1. Background

DoD relies on an ever expanding inventory of expensive and complex software that needs to be maintained. Maintenance costs are significant. They can account for between 50 and 75% of the software life cycle cost [Boehm 76, Swanson 80]. Accordingly, planning for software maintenance must begin early in the software acquisition process. This includes identifying the technology which will be needed to maintain the system and planning to acquire adequate intellectual property rights in that technology.

Our field research has revealed that many of the individuals involved in software acquisition view software maintenance as substantially similar to hardware maintenance. This orientation often fails to appreciate the complexity of software maintenance, and the continuing importance of acquiring the technology needed to maintain the software. To gain this kind of appreciation, it is necessary to understand the unique characteristics of software maintenance as well as the role maintenance plays in the software life cycle.

1.1 Software Maintenance and Enhancement

Software maintenance has been defined as "the process of modifying existing operational software while leaving its primary functions intact" [Boehm 81, p. 54]. This process can be broken down into three functions: 1) correcting an error or "bug" which is found after the software has been placed in operation (corrective maintenance); 2) improving the software to respond to new needs or desires of the user (perfective maintenance); and 3) changing the software to adapt to environmental changes such as a new operating system or new hardware (adaptive maintenance) [Swanson 80]. Studies have shown that approximately 42 to 60% of activities referred to as software maintenance are actually activities used to improve the system to meet new user needs and desires [Swanson 80]. Thus, a substantial proportion of the activities generally termed software "maintenance" is in fact software "enhancement."

In this respect, software maintenance is fundamentally different from hardware maintenance. Hardware maintenance generally involves servicing or replacing deteriorated components and stocking spare parts. In contrast, software maintenance not only corrects defects but also improves functionality. Thus, while hardware maintenance is intended generally to return the hardware to its original functioning level, software maintenance often involves adding new capabilities.

Maintaining software is further complicated by the interdependence among software modules. A repair to one part of the software may affect its other parts and later emerge as a bug in the system [Martin 83]. Although there are certain design methods which can reduce this ripple effect, there is little probability that it can be eliminated.

The dynamics of software maintenance have led several scholars to characterize it as evolutionary. Thus, at least for larger systems, a client's first set of requirements is seldom its last. DoD personnel have told us that this is often true of mission critical systems.

When viewed in this light, the term "maintenance," with its hardware connotations, is often misleading, because it does not recognize the need for continuing development of software. Taking this factor into account, the DoD Joint Logistics Commanders' Workshop on Post Deployment Software Support adopted the term "Post Deployment Software Support" (PDSS). PDSS is defined as:

> "the sum of all activities required to ensure that, during the production/deployment phase of a mission critical computer system's life, the implemented and fielded software/system continues to support its original operational mission and subsequent mission modifications and product improvement efforts" [JLCW 84, p. 4-1-5].

This definition moves away from a "hardware orientation" in that it reflects an appreciation of the complexities of software maintenance. Accordingly, this report will use the term "software support" to refer to both software maintenance and enhancement.

1.2 Software Support Is a Phase of the Software Life Cycle

Software support does not occur in a vacuum; it is a phase in the life cycle of the software. The software life cycle has been defined as "the entire process, from beginning to end, of the development and use of software" [Glass 81, p. 5]. The software life cycle concept was created to help classify the various activities required to develop and maintain software.

There is no definitive life cycle model; different models emphasize different aspects of the cycle [Fairley 85, p. 37]. An example of a typical life cycle model, adapted from the work of Dr. Richard Fairley, is as follows:

1. Analysis. This phase has two parts: a) planning and b) defining the requirements. Planning involves understanding the customer's problem; performing a feasibility study; developing a solution; and planning development. Definition of requirements pinpoints the basic functions of the software component in a system which includes hardware, software and supporting personnel, and produces a specification.
2. Software Design. This phase identifies software components, defines their relationships, and specifies software structure. It also records design decisions and provides a blueprint for the next phase, implementation.
3. Implementation. This phase involves translating design specifications into source code, debugging, documentation, and unit testing.

4. System Testing. System testing involves two kinds of activities: integration testing and acceptance testing.
5. Support. This phase involves enhancing software capabilities, adapting the software to environmental changes and correcting bugs in the system.

The life cycle yields a more realistic view of software costs by focusing on the total lifetime of the software, rather than on each separate phase. "Assessments of the economic viability of a program must include *total lifetime costs* and their life cycle *distribution*, and not be based exclusively on the initial development costs" [Lehman 83, p. 200]. Because support accounts for a large percentage of costs in the total software life cycle, it is important to plan for support in the early stages of an acquisition. This entails not only designing the system to anticipate future support needs, but also identifying the technology for these needs.

1.3 The Sophisticated Nature of Software Support Requires a Transfer of System Expertise

As noted in the foregoing section, due to the broad scope and complexity of software support, its use requires a higher level of expertise than its hardware counterpart. Because such a large portion of support is devoted to enhancing the system, many scholars have characterized it as a microcosm of the development process [Glass 81, Swanson 80, Fairley 85]. To ensure that a system will remain viable and responsive to users' needs throughout the support phase, the system's developer must transfer its expertise to personnel who will support the system.

2. Software Support Requirements

Structuring a data rights policy that will meet DoD's software support needs requires an understanding of the technology needed for software support. In our field research, many people involved in software support stressed that support personnel need to understand the developer's activities during the earlier phases of the software life cycle to perform their jobs adequately. The primary means for communicating this information between the developer and the support personnel are through the source code, software documentation and training. Additionally, support personnel often need access to development tools as well as the latest support management tools. A potential data rights conflict exists in this area, however, because this technology often contains information that the developer considers proprietary.

2.1 Source Code

"Source Code" refers to instructions, written in a programming language, that cause the computer to perform some desired function. A person who understands the programming language can generally understand these instructions. Since the source code is the man readable form of the software and is the form in which the software is developed and supported, access to the source code is vital to software support.

2.2 Computer Software Documentation

Another important means of communicating information about the software to support personnel is through documentation. The source code itself is a form of software documentation. Other documents identified by DoD support personnel as relaying this information include: 1) training and user manuals; 2) specifications prepared for the software development; 3) documents relating to the design of the software; 4) documents listing data gathered from testing and validation procedures; and 5) configuration management documentation, which keeps track of changes to the software.

Software documentation may exist in hard copy or electronic form. For example, it could be delivered in machine readable form, on a diskette, or on a tape. Documentation may also be received from the vendor over a computer terminal.

Both our interviews with DoD personnel and other studies in this area indicate that, despite its significance to the support effort, software documentation is often poor in quality and it does not adequately convey information to support personnel [McCall 83, Swanson 80, GAO 81]. Since documentation is used to train new support personnel, training is severly hampered when the documentation is inadequate.

Access to software documentation has reportedly been a problem for the DoD, particularly with respect to hard copy documentation. Some DoD personnel felt that electronic documentation might resolve some of these difficulties. The quality of the software documentation, as well as the ease with which it can be obtained, affect its usefulness for software support, and therefore it should be considered in developing a regulatory software support policy.

2.3 Software Tools

Government support personnel also identified software tools as a vital element of software support. "Software tools are computer programs which can be used in the development, analysis, testing, maintenance, and management of other computer programs and their documentation" [Osborne 83, p. 44]. These are often complex, sophisticated programs which may need to work in conjunction with other programs [Boehm 81, p. 463-66]. They can require the investment of substantial resources. *Id.*

There is considerable interest among software engineers in developing software tools for the various phases of the life cycle in order to improve productivity [Barbacci 85]. Because these tools help standardize activities performed at various phases of the life cycle, many of our interviewees felt that increased use of these tools will help support software. Indeed, numerous software tools used in development are considered essential to software support. Others aid that support by capturing and storing documentation and configuration management information [JLCW 84, p. 4-4-28 - 4-4-33]. Difficulties in obtaining access to software tools, as well as documentation needed to support those tools, were major areas of concern to DoD personnel. In the absence of gaining access to these tools and to related documentation, there is a feeling that DoD becomes increasingly locked into a "sole source support" relationship with the original developer.

However, the software tools DoD needs in these situations are often among the most innovative of technological developments and involve substantial private investment. Developers therefore tend to be reluctant to use proprietary software tools to perform DoD contracts, because DoD may attempt to claim rights in those tools under standard data rights clauses [SDRC 85]. Indeed, the present data rights policy discourages even the development of these tools because of concerns relating to proprietary rights.

2.4 Training

Due to the high level of technical sophistication needed to understand complex system software, transferring expertise about the system requires more than merely transferring the source code, software documentation, and software tools to the support personnel. Numerous interviewees suggested that training of support personnel is also vital to successful software support. This training can take place informally between the developer and support personnel while the system is being developed. It can also be accomplished by formal training courses. Accordingly, training requirements, like requirements for documentation and support tools, must be identified early in the acquisition process.

3. Variables Affecting Transfer of Technology

Recognizing that support is a significant phase in the software life cycle, DoD has begun to plan for this aspect of the life cycle in the early stages of an acquisition. As discussed above, an integral part of this planning is identifying the software tools, documentation, and training needed for support. Our field research revealed that support requirements often vary from acquisition to acquisition and that the most significant variable relates to the method of carrying out software support. Other significant variables are standardization, use of higher order programming languages, and reusability.

Understanding these variables may enable DoD to tailor an acquisition to its specific technical support needs. This kind of tailoring requires a flexible software/data rights acquisition policy that would allow DoD to limit intellectual property claims to those expressly needed for support requirements. The flexibility would, in turn, considerably ease the tension between industry and DoD on data rights issues.

3.1 Choice of Group to Carry Out Support

One of the most important variables affecting a system's technological needs is who will actually carry out the software support. The choice of who will perform the support will help isolate the technology the original developer needs to transfer to support personnel. For example, our research indicates that involving the original developer in support will often lessen or eliminate the need to transfer technology.

One major factor that may affect the choice of who will carry out the support is the Competition in Contracting Act, 10 U.S.C. 2300 *et seq.* (CICA) [CICA 84]. CICA requires that a support contract be reopened for competition periodically, unless the criteria justifying a sole source procurement are met. 10 U.S.C. 2304. For some systems, this may mean that the support will be performed by a series of different contractors over the system's lifetime. Therefore, a mechanism must exist to transfer system expertise from the developer to subsequent support contractors. Our research revealed, however, that contracts for software support are more likely to meet CICA's sole source criteria than are other kinds of procurements. See 10 U.S.C. 2304(f). Thus, when planning for software support it is important to consider whether competition for software support will be required by CICA, a determination which will ultimately influence the degree to which technology will be needed to be transferred to DoD for future support.

The Project's field investigations revealed three basic models within DoD that may be involved in support after initiation of the system: 1) an organic support activity; 2) the original developer contractor; and 3) the independent contractor.

3.1.1 Organic Support Activity

An organic support activity refers to a group which assumes the responsibility for managing a software system once it has been delivered to a particular service. That activity reports to the system project manager. Because it serves as the focal point for the transfer of technology from the developer to the government, it is essential that the support activity be identified early to enable the support group to acquire the personnel, equipment, and training needed for the new system.

Our interviews indicated that this activity usually attempts to use a mix of military, civil service, and contractor personnel to support a system. This mix stems from the government's desire to retain technical and managerial control over the software support effort. Some of our interviewees were concerned about preventing the expertise in a certain system from being retained exclusively by the contractor, thereby causing the government to lose control over the support effort. Other interviewees discussed various optimum ratios between government personnel and contractors; for the most part, it was determined that at least 20 to 30% of the support group should be government personnel.

Many of our interviewees experienced difficulty in attracting and retaining experienced, qualified civilians to perform within and/or manage the organic support group. The shortage of qualified software engineers is not, of course, limited to DoD, but DoD's problem is exacerbated by hiring limits, long lead times in hiring, and its inability to offer salaries competitive with industry. These interviewees suggested that the problem of attracting qualified support personnel could result in contracting out more support work to private industry.

The organic support group can use either personnel from the developer of the software system or other independent contractors to assist government personnel. In those instances where only personnel from the developer are used, the developer's expertise would be transferred solely to government personnel. However, where competition or other factors mandate the participation of non-developer contractor personnel, both government and contractor personnel will need access to the training, documentation, and tools necessary to support part or all of the system.

Thus, certain organic support groups may require transferring system expertise to a wide variety of civilian, military, and contractor personnel over a system's life. It is therefore not surprising that the factors most frequently cited by the interviewees as essential for good organic support were good software tools and documentation, involvement of government personnel during the system's development, and a good training program for support personnel. If the technical requirements to support a system organically are evaluated carefully on a system by system basis, the government should be able to gain access to the needed technology through appropriate legal mechanisms.

3.1.2 Support by Original Developer

The system project manager may contract with the original developer of the software system to perform software support under the manager's direction. This strategy could either provide for long term support or it could be a short term project designed simply to enable the government to transfer expertise about a project to its own support staff.

The JLC PDSS Workshop Panel on Industry Government Work Force Mix concluded that the need for the original developer's participation is always greater in cases involving complex and immature software with the need diminishing substantially as the software matures [JLCW 84]. In fact, interviewees frequently referred to existing contracts which provide for the developer to afford the support until the organic support group is able to take over the system. Arrangements of this nature generally last from two to three years.

With regard to long term support agreements, we were informed of a support contract with the original developer which lasts as long as fifteen years. Under a contract of this type, the developing contractor effectively becomes responsible for the system over its entire life cycle. When the original developer maintains the system, there appears to be little need to transfer system expertise to the government. However, a number of contingencies should be anticipated. The government should insert safeguards into the contract to protect the government when a developer: 1) does not perform satisfactorily; 2) loses interest in supporting the system; or 3) goes out of business.

Several interviewees noted that there is often inadequate documentation to allow for support of certain software systems, especially large and complex ones, and that the requisite knowledge and experience frequently resides exclusively within the developing company. This problem has led DoD in at least one reported situation to enter into a long term support contract for a major system.

Different interviewees discussed cases in which weapons systems had been maintained initially by the developing contractor under a sole source contract, but when the government was unable to justify the sole source arrangement under CICA, it had to open the support work to competitive bidding. In these cases, the government awarded the contract to an independent contractor. In at least one instance, the independent contractor was unable to perform the support and the system ceased to function. The original developer was rehired to resume support of the system. In another case, government personnel stressed the high cost of acquainting a non-developer contractor with a new system.

3.1.3 Support by Independent Contractor

Under this scenario, a contractor who is not the original developer enters into an agreement with the system project manager to support the system. This arrangement usually stems from CICA's competitive bidding requirements. The transfer of the original developer's technology through training and documentation is critical to the success of this arrangement. Thus, technical support requirements must be identified early and a method to gain access to the developer's technology must be provided. In some instances, this transfer of technology will occur as the result of natural market forces. For example, if contractor "A" develops a system, but does not obtain the support contract for it, many of A's employees who have worked on the system often migrate over to contractor "B," who has obtained the support contract. If, three years later, competition is reopened for the support contract and contractor "C" wins, some of these same employees will migrate to contractor "C." Thus, independent market forces keep experienced technicians working on the system, but the natural inefficiencies of these forces often require the government to pay a premium for the support effort.

3.2 Technological Variables

3.2.1 Standardization

Many of the interviewees indicated that an increased interest in standardizing computer resources exists within DoD. The government may impose standards in areas such as hardware, interfaces, methodology, programming language, or documentation. This has been an area of considerable interest to the Navy, and, to some extent, to the Army and the Air Force [GAO 81]. For example, a standard may require that software be compatible with a particular computer. Other standards may require that the software product can be used with another computer program or system. This type of software standard is called an interface standard [Morton 86].

A standard may require use of a particular programming language or a certain methodology. Use of the Ada language for programming is an attempt to standardize DoD "mission critical" software. Similarly, DoD Standard 2167 requires contractors to provide documentation at specified intervals and in a particular form [Golubjatnikov 86, STD-2167 84]. By standardizing the programming language and documentation requirements, DoD may improve communication between the developer and support personnel, thus improving the government's ability to support its own system. Further, in interviews with DoD personnel, the Project learned that standardization encourages private competition for the support contract, because it avoids a unique, non-standard technology which only the original developer understands and can support.

However, our interviews revealed that standardization is a double-edged sword. Since standardization places constraints on development, DoD could become locked into a particular technology or approach at the expense of availing itself of later developments. Thus, standardization may require a trade-off between obtaining the latest technology and imposing standards that may, in turn, allow for better support, in: operability, and reusability.

3.2.2 Programming Language

DoD is moving toward requiring that the Ada language be used for higher order programming in "mission critical" software unless a waiver can be obtained [DoD-D5000.31 83, Pellerin 86]. This type of higher order language enables the developer to write the source code at a higher level of abstraction with fewer instructions required for each desired function. It generally aims to provide a logic or structure for programming at a more understandable level. The goal of this higher level of abstraction is to improve the productivity of the programmer. Additionally, the code itself may be easier to read by one skilled in the use of the programming language [Glass 81, p. 93-94]; [Sammet 86].

A major benefit of programming in a higher order language is that making the source code easier to read tends to improve communication between the developer and support personnel, and thus improves software support. It also decreases the need for other supporting documentation, thereby reducing the likelihood of a data rights dispute between the government and the developing contractor.

3.2.3 Reusability

Reusability is a concept which is gaining attention in both DoD and the software industry generally. Software reuse generally refers to reusing sections of code, software design, or some product of a software life cycle in a separate software system [Jones 84, Horowitz 84]. Reusability is a key to improving productivity in the software development area [Silverman 85, IEEE 84, Barbacci 85].

As we learned in our research, one example of software reuse within DoD is the Common Ada Missile Package Program (CAMP). Further, some interviewees characterized DoD's increasing use of commercial off-the-shelf software (COTS) and non-developmental items (NDI) as forms of software reuse. Similarly, "the entire spectrum of commercially marketed systems and application programs" [Jones 84] is also a form of software reuse.

Software reuse requires the following: 1) technological expertise; 2) a storage, distribution and accessing system for the items to be reused; and 3) resolution of the intellectual property and data rights issues raised by the use of the work of an additional developer [Silverman 85, Horowitz 84, TR-1 86, Ch. 4]. As our interviews with DoD personnel indicated, software reuse may affect software support. If a proprietary module is reused DoD may not be given access to the supporting documentation and software tools. In such a case, DoD may have to rely upon the developer who holds the proprietary interest in the reused item to support the software system.

4. Decision Making Structures Within DoD

As discussed above, the technical requirements for software support are governed by a broad spectrum of variables. While the system is still being acquired, the government should evaluate the impact of these variables on future needs for software tools, documentation, and training. Our field research revealed that, in order to evaluate these variables effectively, the program manager must consult both the personnel who will oversee support and the user of the system. Although the services are beginning to recognize the importance of consulting user and support personnel, there is still a need for improved communications among these groups.

The mechanisms for obtaining input from user and support personnel vary from program to program and from se :ce to service. This takes place in a highly complex, politically charged environment in which key decisions affecting the future life of a system are made.

4.1 The Role of the Program Manager

The interviews with DoD personnel indicated that the program manager is generally the central figure in development and procurement. The program manager oversees development, and is responsible for budgeting, scheduling, and quality.

Our interviews revealed that the program manager tends to be motivated by incentives which differ from those of support and user personnel. On the one hand, support personnel are concerned with obtaining access to the software tools, documentation, and training necessary to maintain a system. On the other hand, the program manager tends to be evaluated on whether he or she has the system completed within scheduling and budgetary constraints. Very often, when these constraints are not being met, the manager sacrifices the tools or documentation needed for maintenance. Thus, the support and user personnel need to have a voice in the development process so that their needs are adequately addressed.

4.2 The Role of User Personnel and Support Personnel in Initial Planning for a Software System

4.2.1 Need for Early Participation

DoD personnel indicated that there is a need to consult support and user personnel early in the development of the software. The two primary reasons expressed for this need were: 1) support is improved by anticipating future support needs early in the software life cycle; and 2) software design issues tend to be frozen early in the development process. Additionally, the program budget must take into account the costs of software tools and documentation.

We describe below mechanisms which have been developed to allow consultation with support and user personnel.

4.2.2 Mechanisms for Ensuring Participation by User and Support Personnel

Involvement by user and support personnel in the initial development of the software occurs through completion of a document referred to as the Computer Resources Life Cycle Management Plan (CRLCMP).[2] The CRLCMP identifies the group that will support the software. Additionally, the CRLCMP should address the items needed for support, such as software tools and documentation. The CRLCMP is developed by a working group which includes the program manager, representatives of support and user personnel, and other interested parties.

During the initial stages, the program manager directs the development of the CRLCMP. In theory, the CRLCMP is developed by consensus, although the program manager retains some discretion. Moreover, the relationship between the program manager and the user/support personnel may be complicated by differences in physical location and in chain of command. Thus, the actual impact of support and user personnel on the development of the CRLCMP may vary from service to service and from program to program. Some interviewees indicated that the CRLCMP is not always given a sufficiently high priority during an acquisition.

Project interviews also indicated that support and user personnel generally have an opportunity to comment on support issues at progress meetings conducted at various phases of the development process. As with the CRLCMP, however, the program manager has discretion in responding to those comments. There exists, therefore, consid-

[2]The term CRLCMP seems to be in general use with the Air Force, and to an increasing extent with the Navy. The comparable document currently used by the Army is referred to as the Computer Resources Management Plan (CRMP).

erable variation among the services, and among programs, as to the degree that support and user personnel actually affect the software development/acquisition process.

Thus, the mechanisms for ensuring participation by user and support personnel in the planning process already exist or are being developed. If DoD fails to use these mechanisms properly, it may risk requesting unneeded technology and/or not seeking technology needed for software support.

5. Conclusion

This report is designed as a guide for lawyers and policymakers who are involved in the transfer of proprietary knowledge from the private to the public sector. Software support is a significant and sophisticated undertaking requiring a transfer of system expertise and tools from the developer to support personnel. The challenge lies in crafting a software acquisition/data rights policy which will be effective in transferring technology and expertise, while retaining the incentive to private industry to create new technologies. Understanding the technology needed for software support, the variables which may affect support requirements, and the program needs of user and support personnel should aid the government in tailoring support needs to those of a particular acquisition. A flexible software acquisition/data rights poli which is responsive to technological needs could be a major step toward striking wha e Packard Commission called the "delicate and necessary balance" between DoD and industry.

APPENDIX A

Facilities Visited within the Department of Defense:

ARMY

Army Materiel Command, HQ
Alexandria, VA.
    Battlefield Information/C4 Division

Information Systems Engineering Command
Fort Belvoir, Virginia

Communications and Electronics Command
Fort Monmouth, New Jersey
    Center for Life Cycle Software Engineering
        Intelligence/EW Division
        Communications Division
        Software Support & Development Division
    Office of the Competition Advocate

NAVY

Office of Chief of Naval Operations (OPNAV 945)
Washington, D.C.
    Information Systems Division, Space Command and Control

Navy Space and Warfare Command
Arlington, VA.
    Warfare Systems Engineering Policy Division

Naval Surface Weapons Center
Dahlgren, VA.
    Combat Systems Department
        AEGIS Ship Combat Systems Division
        Combat Systems Design & Engineering Division
        Cruise Missile Weapon Systems Division
        AEGIS Program Office
    Electronic Systems Department
        Command Support Systems Division
    Office of Counsel

Naval Ocean Systems Command
San Diego, California (SEI Resident Affiliate)

AIR FORCE

Electronic Systems Division
Hanscom Air Force Base, Massachusetts
    Program Office CCPDSR System
    Program Office CSSR System
    Logistics Command Liasion Office
    Office of the Competition Advocate

Air Force Logistics Command
Wright-Patterson Air Force Base, Ohio
  Computer Systems Staff, Computer Resources Branch
  Air Force Acquisition Logistics Center
  Office of the Competition Advocate
  Information Systems Laboratory

Air Force Systems Command
Andrews Air Force Base, Maryland
  Product Assurance and Acquisition Logistics Div.

APPENDIX B: Bibliography

[Barbacci 85] Mario R. Barbacci, A. Nico Habermann and Mary Shaw.
The Software Engineering Institute: Bridging Practice and Potential.
IEEE Software :4-21, November, 1985.

[Boehm 76] Barry W. Boehm.
Software Engineering.
IEEE Transactions on Computers C-25(12):1226-1241, December, 1976.

[Boehm 81] Barry W. Boehm.
Software Engineering Economics.
Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1981.

[Brooks 82] Frederick P. Brooks, Jr.
The Mythical Man-Month.
Addison-Wesley, 1982.

[Buckley 86] F. J. Buckley.
An Overview of the IEEE Computer Society Standards Process.
Proceedings Computer Standards Conference 1986 :2-8, May, 1986.

[Buzzard 85] G.D. Buzzard and T.N. Mudge.
Object-Based Computing and the Ada Language.
Computer 18(3(ISSN 0018-9162)):11-19, March, 1985.

[CICA 84] Competition In Contracting Act of 1984, Pub. L. 98-369 10 U.S.C. Sec. 2301.
1984.

[CSC 86] IEEE Computer Society.
Proceedings Computer Standards Conference 1986.
IEEE, Inc. :2-167, May, 1986.

[Day 85] Raymond Day.
A History of Software Maintenance of a Complex U.S. Army Battlefield Automated System.
Proceedings of the Conference on Software Maintenance , November, 1985.

[DeRoze 78] Barry C. DeRoze and Thomas H. Nyman.
The Software Life Cycle - A Management and Technological Challenge in the Department of Defense.
IEEE Transactions on Software Engineering SE-4(4):309-318, July, 1978.

[DoD-D5000.31 83]
Computer Programming Language Policy, DoD Directive 5000.31.
1983.

[Fairley 85] Richard Fairley.
Software Engineering Concepts.
McGraw-Hill Book Company, 1985.

[GAO 81] General Accounting Office.
Federal Agencies Maintenance of Computer Programs: Expensive and undermanaged.
NTIS PB81-23S020 FMD-81-25. Washington, D.C., 1981.

[Glass 81] R. L. Glass and R.A. Noiseux.
Software Maintenance Guidebook.
Prentice-Hall, Englewood Cliffs, NJ, 1981.

[Golding 86] D.H. Golding, F. Szantai, and M.E. Perrin.
The Adaptable Systems Methodology (A Dynamic Approach to Quality Products).
Proceedings Computer Standards Conference 1986 (698):81-85, May, 1986.

[Golubjatnikov 86] Ole Golubjatnikov.
DOD-STD-2167 Software Development Standards Package Development and Open Issues.
Computer Standards Conference 1986 :1-37, May, 1986.

[Horowitz 84] E. Horowitz and J.B. Munson.
An Expansive View of Reusable Software.
IEEE Transactions on Software Engineering Se-10(5):477-487, September , 1984.

[IDA 84] Institute For Defense Analysis Computer and Software Engineering Division.
Report of The Rights in Data Technical Working Group.
Prepared for Office of the Undersecretary of Defense for Research and Engineering IDA Record Document D-52. I984.

[IEEE 84] A Publication of the IEEE Computer Society.
IEEE Transactions on Software Engineering.
IEEE SE-10(5):473-609, September, 1984.

[JLCW 84] *Final Report of the Joint Logistics Commander's Workshop on Post Deployment Software Support for Mission Critical Computer Software*
Volume II edition, Volume II - Workshop Proceedings, 1984.

[Jones 84] T.C. Jones.
Reusability in Programming: A Survey of the State of the Art.
IEEE Transactions on Software Engineering SE-10(5):488-494, September, 1984.

[Knirk 86] D. L. Knirk.
Reducing Documentation in Software Development:Freedom from Standard Document Definitions.
Proceedings Computer Standards Conference 1986 (698):127-135, May, 1986.

[Lehman 83] Meir M. Lehman.
Programs, Life Cycles and the Laws of Software Evolution.
Proceedings of the IEEE.
reprinted in Girish Parikh and Nicholas Zvegintzov, "Tutorial on Software Maintenance", 1983, pages 1226-1241.

[Lientz 78] B. P. Lientz, E.B. Swanson and G.E. Tompkins.
Characteristics of Applications Software Maintenance .
Communications of the ACM 21(6), June, 1978.

[Martin 83] James Martin and Carma McClure.
Software Maintenance: The Problem and It's Solutions.
Prentice-Hall, Inc., 1983.

[McCall 83] James McCall and Mary Anne Herndon.
Software Maintenance Survey.
Special Report I NBSL:83-SR-035, National Bureau of Standards Institute for Computer Sciences & Technology, June, 1983.

[Morton 86] R. Morton and S.T. Redwine, Jr.
Information Interface Standards for Software Engineering Environments.
Proceedings Computer Standards Conference 1986 (698):42-48, May, 1986.

[Nash 86] S.H. Nash and S.T. Redwine, Jr.
A Map of the World of Software-Related Standards, Guidelines, and Recommended Practices.
Proceedings Computer Standards Conference 1986 (698):136-151, May, 1986.

[NBSIR 82] U.S. Department of Commerce.
A Taxonomy of Tool Features for the Ada Programming Support Environment (APSE)*
final report NBSIR 82-2625 edition, Department of Defense, Arlington, VA , 1982.

[Osborne 83] Roger J. Martin and Wilma M. Osborne.
Guidance on Software Maintenance.
National Bureau of Standards Special Public 500-106.
1983

[OSDStudy 84] OSD Technical Data Rights Study Group.
Who Should Own Data Rights: Government or Industry? Seeking A Balance.
A Report Prepared for the Undersecretary of Defense Research and Engineering. 1984.

[Packard 86] President's Blue Ribbon Commission on Defense Management.
A Quest for Excellence.
Government Report. 1986.

[Parikh 83]	Girish Parikh and Nicholas Zvegintzov. Tutorial on Software Maintenance. *IEEE Computer Society* 453(82-83405), 1983.
[Pellerin 86]	Cheryl Pellerin. Ada Industry Awaits DoD Implementation Standards. *Washington Technology* 1(17):1 and 11, November, 1986.
[RCGD 80]	Requirements Committee, Government Division. *The DoD Digital Data Processing Study - A Ten Year Forecast* Electronic Industries Association, 1980.
[Robinson 86]	G. S. Robinson. Accredited Standards Committee for Information Processing Systems,X3. *Proceedings Computer Standards Conference 1986* (698):9-11, May, 1986.
[S.I. 86]	S. I. Sherr. IEEE and Information Systems. *Proceedings Computer Standards Conference 1986* (698):14-15, May, 1986.
[Sammet 86]	Jean E. Sammet. Why Ada is not just another Programming Language. *Communications of the ACM* 29(8):772-732, August, 1986.
[SDRC 85]	Department of Defense Federal Acquisition Regulation Supplement Parts 52 and 27. 48 CFR Parts 252 and 227. 1985.
[Sherr 86]	S. I. Sherr. ANSI and Information Systems. *Proceedings Computer Standards Conference 1986* (698):12-13, May, 1986.
[Silverman 85]	Barry G. Silverman. Software Cost and Productivity Improvements: An Analogical View. *IEEE Computer Journal* :86-95, May, 1985.
[SOD 80]	The U.S. General Accounting Office. *The Department of Defense's Standardization Program for Military Computers - A more Unified Effort is Needed* LCD-80-69 edition, Washington, D.C., 1980.
[STD-2167 84]	Department of Defense. *Defense System Software Development* Military Standard edition, Washington, D.C., 1984.
[Swanson 80]	Bennet P. Lientz and E. Burton Swanson. *Software Maintenance Management.* Addison-Wesley Publishing Company, 1980.

[TaskForce 84] Software Rights in Data Task Force .
Proposed Reform of Government Rights in Data Clauses.
prepared for use and consideration of Data Rights Group of the Office of the Secretary of Defense. 1984.

[TM1 86] Pamela Samuelson.
Adequate Planning for Acquiring Sufficient Documentation about and Rights in Software to Permit Organic or Competitive Maintenance
SEI-86-TM-1 edition, Carnegie Mellon University Software Engineering Institute, Pittsburgh, Pa., 1986.

[TM2 86] Pamela Samuelson.
Comments on the Proposed Defense and Federal Acquisition Regulations
SEI-86-TM-2 edition, Carnegie Mellon University, Software Eng[illegible]eering Institute, Pittsburgh, Pa., 1986.

[TM3 86] Pamela Samuelson.
Understanding the Implications of Selling Rights in Software to the Defense Department: A Journey through the Regulatory Maze
SEI-86-TM-3 edition, Carnegie Mellon University Software Engineering Institute, Pittsburgh, Pa., 1986.

[TR-1 86] Pamela Samuelson.
Toward a Reform of the Defense Department Software Acquisition Policy
CMU/SEI-86-TR-1 (ESD-TR-86-202) edition, Carnegie Mellon University, Software Engineering Institute, Pittsburgh, Pa., 1986.

[TR-2 86] Pamela Samuelson, Kevin Deasy and Anne C. Martin.
Proposal for a New 'Rights in Software' Clause for Software Acquisitions by the Department of Defense
CMU/SEI-86-TR-2 (ESD-TR-86-203) edition, Carnegie Mellon University, Software Engineering Institute, Pittsburgh, Pa., 1986.

[VanHorn 83] Earl C. VanHorn.
Software Must Evolve.
Proceedings of the IEEE.
reprinted in Girish Parikh and Nicholas Zvegintzov, "Tutorial on Software Maintenance, 1983.
reprinted in IEEE Computer Society.

[VanNostrand 83] Anthony Ralston and Edwin D. Reilley, Jr. (editor).
Encyclopedia of Computer Science and Engineering.
VanNostrand Reinhold Company, New York, 1983.

[Wade 85] James P. Wade, Jr. Assistant Secretary of Defense for Acquisition and Logistics.
DOD Acquisition Improvement - The Challenges Ahead: Perspectives of the Assistant Secretary of Defense for Acquisition and Logistics.
1985.

[Winthrop 82] Girish Parikh.
Techniques of Program and System Maintenance.
Parikin and Winthrop, Chicago, IL, 1982.

[Wood 86] H.M. Wood.
Emerging Software Standards: Opportunity and Challenge.
Proceedings Computer Standards Conference 1986 (698):122-126, May, 1986.

AD-A178971

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b. RESTRICTIVE MARKINGS: NONE
2a. SECURITY CLASSIFICATION AUTHORITY: N/A	3. DISTRIBUTION/AVAILABILITY OF REPORT: Unlimited, Unclassified distribution
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): SEI-87-TR-2	5. MONITORING ORGANIZATION REPORT NUMBER(S): ESD-TR-87-102
6a. NAME OF PERFORMING ORGANIZATION: SOFTWARE ENGINEERING INST. — 6b. OFFICE SYMBOL (If applicable): SEI	7a. NAME OF MONITORING ORGANIZATION: SEI JOINT PROGRAM OFFICE
6c. ADDRESS (City, State and ZIP Code): CARNEGIE-MELLON UNIVERSITY PITTSBURGH, PA 15213	7b. ADDRESS (City, State and ZIP Code): ESD/XRS1 HANSCOM AIR FORCE BASE HANSCOM, MA 01731
8a. NAME OF FUNDING/SPONSORING ORGANIZATION: SEI JOINT PROGRAM OFFICE — 8b. OFFICE SYMBOL (If applicable): ESD/XRS1	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State and ZIP Code): CARNEGIE-MELLON UNIVERSITY PITTSBURGH, PA 15213	10. SOURCE OF FUNDING NOS.

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.
63752F	N/A	N/A	N/A

11. TITLE (Include Security Classification) The Effect of Software Support Needs on the Department of Defense Software Acquisition Policy: Part 1 A Framework for Analyzing Legal Issue

12. PERSONAL AUTHOR(S) Anne C. Martin, Kevin M. Deasy

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Yr., Mo., Day)	15. PAGE COUNT
FINAL	FROM ______ TO ______	January 1987	34

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB GR.	

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

It is often in the context of acquiring intellectual property needed to maintain and enhance software that data rights disputes arise between DoD and private industry. DoD representatiives identify access to source code, documentation and software tools as being vital to maintenance and enhancement. This report discusses technical and managerial variables that might affect DoD's need for such intellectual property. It is concluded that the choice of a group to carry out maintenance and enhancement is a key factor in determining DoD's requirements. Other important variables include use of software standard:, higher order programming languages and reused software. Consideration of these factors could enable DoD to tailor more narrowly an acquisition to those items needed to maintain and enhance software, potentially reducin both DoD's costs, and the tension that exists between DoD and private industry with respect to data rights issues.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT: [✓ UNCLASSIFIED/UNLIMITED] SAME AS RPT. ☐ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION: UNCLASSIFIED, UNLIMITED DISTRIBUTION
22a. NAME OF RESPONSIBLE INDIVIDUAL: KARL H. SHINGLER	22b. TELEPHONE NUMBER (Include Area Code): 412 268-7630 — 22c. OFFICE SYMBOL: SEI JPO

DD FORM 1473, 83 APR — EDITION OF 1 JAN 73 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE

END
4-87
DTIC